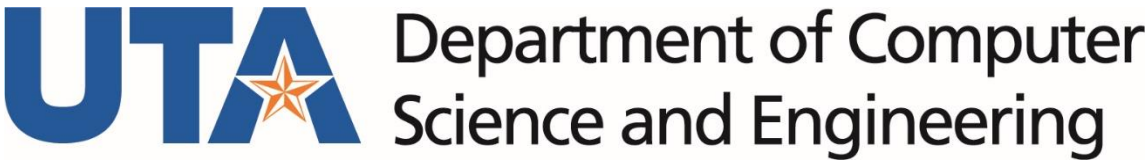
CSE 1320




# CSE 1320 Project Documentation

## The Universal Unit Converter

Students' names, surnames, IDs:

1. Subhaan Elburz, 1002135522
2. Michael Tran, 1002098656

Mentor: Dr. Marika Apostolova
TA: Charlie Vuong

CSE 1320


**Intermediate programming CSE 1320**

**Student declaration:**

*We declare that:*

- *We understand what is meant by plagiarism.*
- *The implication of plagiarism has been explained to me by our professor.*
- *This assignment is all team own work and we have acknowledged any use of the published and unpublished works of other people.*

**1 Student** name, surname, ID and **signature: Subhaan Elburz, 1002135522, Subhaan Elburz**
**2 Student** name, surname, ID and **signature: Michael Tran, 1002098656, Michael Tran**

**Date: 4/21/24**

| | | |
|---|---|---|
| | **Total number of pages including this cover page** | 22 |
| **Class Code / Group** | CSE 1320 | |
| **Lecturer's Name** | MARIKA APOSTOLOVA | |

CSE 1320

# Table of Contents

CSE 1320

# CHAPTER ONE: PROJECT INTRODUCTION

## Introduction

The project that we created is the Universal Unit Converter (or UUC), a robust and versatile unit converter. As everyone knows, a unit converter is used to convert measurements. Our project allows the users to seamlessly convert between a wide variety of units. This wide range of units includes time, temperature, mass, length, metric units, and imperial units. The unit converter that we created can be used for multiple reasons, whether it's for homework assignments, projects, reports, or even everyday tasks like baking. Our Unit Converter aims to provide a user-friendly interface coupled with accurate and efficient conversions.

## Project Specification/Function Modules

Our project allows the user to effortlessly convert units from a variety of categories including time, length, temperature, and mass. In the code, this is accomplished by creating functions for each unit of measurement, for example, the functions are named Time(), Length(), etc. All of these functions will follow the same format by first printing the list of units, then asking the user the number, the unit they wish to convert from, and the unit they wish to convert to. All of these functions will have the conversion formulas for each unit that it lists. For example, the Time() function lists seconds, minutes, hours, and days, and has all of the conversion formulas between the listed units. Additionally, there is a printHistory() function that will print all of the conversions the user has done in the past. All of these functions are called in the switch statement found in the main function. This will all be explained in more detail in the Design and Codes section.

## Project (Input/Output) Specification

We've made our unit converter extremely user-friendly to guarantee an enjoyable user experience. This is evident through the very straightforward inputs/outputs. When the program first starts, it will open up to our logo and a main menu screen with all the options of measurements to convert. All the user has to do is input the number that represents the unit of measurement they want to convert. For example, the user can enter the number 1 for Time. Then, the program will switch to a screen asking the user to input three things: the value of the conversion, the unit of the original value, and then the unit the user wants to convert to. The program should then output the converted value and return to the original main menu screen.

## Screen Design

In this section, we will preview the visual and interactive elements that make up our user interface. In the images, we can see that the program has first started up, with the logo and main menu being printed first. It prompts the user to select an option, and in this case, the user enters in 2. Then, a second menu pops up with a list of Units and it prompts the user to enter in the value of the number the user is converting, the original unit of that number, and the desired unit to convert to. In this case, the user enters in 300, 3, and 7, which corresponds to 300 Kilometers to Miles. The program then outputs that 300 Kilometers is 186.45 Miles. After this, it returns to the main menu, and the user enters in 5 to print the conversion history. It then prints that the conversion history is: 300 Kilometers is 186.45 Miles. Lastly, it returns back to the main menu, where the user enters in 6, and exits the program.

```
**********************************************
          The Universal Unit Converter
**********************************************
[1] Time
[2] Length
[3] Temperature
[4] Mass
[5] Print Conversion History
[6] Exit
**********************************************
Please select an option: 2
**********************************************

**********************************************
[1] Centimeters
[2] Meters
[3] Kilometers
[4] Inches
[5] Feet
[6] Yards
[7] Miles
**********************************************
Please enter the value of the number you are converting: 300
Please enter the unit you wish to convert from: 3
Please enter the unit you wish to convert to: 7
```

*Figure 1: Test-run-1a*

```
**********************************************
300.00 Kilometer(s) is 186.45 Mile(s)
**********************************************

**********************************************
          The Universal Unit Converter
**********************************************
[1] Time
[2] Length
[3] Temperature
[4] Mass
[5] Print Conversion History
[6] Exit
**********************************************
Please select an option: 5
**********************************************


**********************************************
               Conversion History
**********************************************
300.00 Kilometer(s) is 186.45 Mile(s)
**********************************************

**********************************************
          The Universal Unit Converter
**********************************************
[1] Time
[2] Length
[3] Temperature
[4] Mass
[5] Print Conversion History
[6] Exit
**********************************************
Please select an option: 6
**********************************************

**********************************************
You have exited the application. All conversions have been saved in the history.txt file.
**********************************************
```

*Figure 2: Test-run-1b*

CSE 1320

# CHAPTER TWO: DESIGN AND CODES

Here, we will break down how the Universal Unit Converter was built, focusing on the design choices and the code behind it. In order to show the logic behind different functionalities, we'll look at code snippets with supporting explanations.

## Printing The Logo

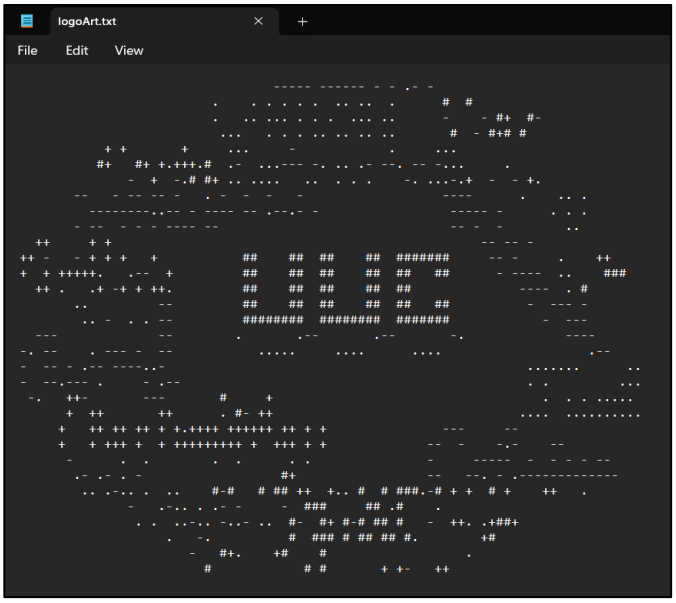



*Figure 3: logoArt.txt-file*

*Figure 4: Printed-logo*

```c
FILE *logo;
char logoString[100];
logo = fopen("logoArt.txt", "r+");
if(logo == NULL){
    printf("Error opening logo file. \n");
    exit(1);
}
while(fgets(logoString, sizeof(logoString), logo) != NULL){
    printf("%s", logoString);
}
```

*Figure 5: Code-to-print-logo*

This code snippet prints out the Universal Unit Converter logo when you first run the program. How this works is that a file pointer, "logo," is created and opens up a file named "logoArt.txt" in reading mode. Inside this text file, there is an ASCII art version of our logo. So, after we open the file, we check if there were any errors, and if there were any errors we will exit the program. If there are no errors, then we will enter a while loop that will print out the ASCII art logo line by line by using the fgets command. In the command, we save each line into a string, "logoString[100]," and it will do so until it reaches the end of the file and prints out the entire ASCII art logo.

## The Main Menu

```
do{
    printf("\n***********************************************");
    printf("\n\t  The Universal Unit Converter \n");
    printf("***********************************************\n");
    printf("[1] Time \n");
    printf("[2] Length \n");
    printf("[3] Temperature \n");
    printf("[4] Mass \n");
    printf("[5] Print Conversion History \n");
    printf("[6] Exit \n");
    printf("***********************************************\n");
    printf("Please select an option: ");
    scanf("%d", &selection);
    printf("***********************************************\n");
    printf("\n");
    switch(selection){
    case 1:
        Time();
        break;
    case 2:
        Length();
        break;
    case 3:
        Temperature();
        break;
    case 4:
        Mass();
        break;
    case 5:
        printHistory();
        break;
    case 6:
        printf("***********************************************\n");
        printf("You have exited the application. All conversions have been saved in the history.txt file. \n");
        printf("***********************************************\n");
        break;
    default:
        printf("***********************************************\n");
        printf("Please choose one of the listed choices. \n");
        printf("***********************************************\n");
        break;
    }
}while(selection != 6);
```

*Figure 6: Code-to-print-main-menu*

```
***********************************************
          The Universal Unit Converter
***********************************************
[1] Time
[2] Length
[3] Temperature
[4] Mass
[5] Print Conversion History
[6] Exit
***********************************************
Please select an option:
```

*Figure 7: Printed-main-menu*

This code snippet prints out the main menu of the Universal Unit Converter. It does this by first going into a do-while loop that prints out the main menu with all of the listed options, as seen in Figure 7. The user will have to enter a number between 1 through 6 in order to proceed. This is done by asking the user for their selection with the scanf command. Then, the user's selection will go into a switch statement that will proceed. In the switch statement, the numbers 1 through 4 will move onto the Unit Conversion Menus, the number 5 will print previous conversions, and the number 6 will exit the program. As long as the user does not type in 6 on the main menu screen, the program will remain running. If the user does not type in a

number between 1 and 6, the program will default and tell the user to choose one of the listed choices. This is done in the code by setting the default in the switch statement to tell the user to choose one of the listed choices (See Figure 8).

```
**********************************************
         The Universal Unit Converter
**********************************************
[1] Time
[2] Length
[3] Temperature
[4] Mass
[5] Print Conversion History
[6] Exit
**********************************************
Please select an option: 7
**********************************************

**********************************************
Please choose one of the listed choices.
**********************************************

**********************************************
         The Universal Unit Converter
**********************************************
[1] Time
[2] Length
[3] Temperature
[4] Mass
[5] Print Conversion History
[6] Exit
**********************************************
Please select an option: 6
**********************************************

**********************************************
You have exited the application. All conversions have been saved in the history.txt file.
**********************************************
```

*Figure 8: Wrong-choice*

## The Unit Conversion Menu

```c
void Mass(){
    history = fopen("history.txt", "a+");
    if(history == NULL){
        printf("Error opening history file. \n");
        exit(1);
    }
    float unit1, unit2, num1, num2, repeat;
    printf("**********************************************\n");
    printf("[1] Pounds \n");
    printf("[2] Kilograms \n");
    do{
        printf("**********************************************\n");
        printf("Please enter the value of the number you are converting: ");
        scanf("%f", &num1);
        printf("Please enter the unit you wish to convert from: ");
        scanf("%f", &unit1);
        printf("Please enter the unit you wish to convert to: ");
        scanf("%f", &unit2);
        printf("**********************************************\n");
        if(unit1 > 2 || unit1 < 1 || unit2 > 2 || unit2 < 1 || unit1 == unit2){
            repeat = 1;
            printf("\n**********************************************\n");
            printf("Please choose one of the listed choices. \n");
            printf("**********************************************\n\n");
        } else{
            repeat = 0;
        }
    }while(repeat == 1);
    printf("\n**********************************************\n");
    if(unit1 == 1 && unit2 == 2){
        num2 = num1/2.205;
        printf("%.2f Pound(s) is %.2f Kilogram(s) \n", num1, num2);
        fprintf(history, "%.2f Pound(s) is %.2f Kilogram(s) \n", num1, num2);
    }else if(unit1 == 2 && unit2 == 1){
        num2 = num1*2.205;
        printf("%.2f Kilogram(s) is %.2f Pound(s) \n", num1, num2);
        fprintf(history, "%.2f Kilogram(s) is %.2f Pound(s) \n", num1, num2);
    }
    printf("**********************************************\n");
    fclose(history);
}
```

*Figure 9: Code-for-mass-conversions*

```
**********************************************
         The Universal Unit Converter
**********************************************
[1] Time
[2] Length
[3] Temperature
[4] Mass
[5] Print Conversion History
[6] Exit
**********************************************
Please select an option: 4
**********************************************

**********************************************
[1] Pounds
[2] Kilograms
**********************************************
Please enter the value of the number you are converting: 225
Please enter the unit you wish to convert from: 1
Please enter the unit you wish to convert to: 2
**********************************************

**********************************************
225.00 Pound(s) is 102.04 Kilogram(s)
**********************************************

**********************************************
         The Universal Unit Converter
**********************************************
[1] Time
[2] Length
[3] Temperature
[4] Mass
[5] Print Conversion History
[6] Exit
**********************************************
Please select an option:
```

*Figure 10: Mass-conversion-menu*

This code snippet prints out the Unit Conversion Menu for Mass Conversions. It first starts by setting the history file pointer to open the “history.txt” file in append mode. It then checks if there are any errors, if there are, then it exits the program. If not, it prints out the menu and enters a do-while loop that will ask the user to enter the value of the number they are converting, the unit they wish to convert from, and the unit they wish to convert to. This is in a do-while loop so that if the user enters a number that is not one of the options, or if the user entered the same units for conversion, it would ask the user to re-enter the details of their conversion (See Figure 11). After the user has successfully entered the three inputs, the program will enter an if statement that has every combination of the conversions between the units. For example, if the user entered 1 (pounds) and then 2 (kilograms), it would go to the if statement where unit1 == 1 and unit 2 == 2. Then, in the if statement, it will do the conversion, print out what the conversion is, and also save the conversion in the “history.txt” file (See Figure 10). After the conversion, the program will close the history file pointer, and return to the main menu. In this case, we picked the Mass conversion mainly because it has the least number of options.

```
**********************************************
          The Universal Unit Converter
**********************************************
[1] Time
[2] Length
[3] Temperature
[4] Mass
[5] Print Conversion History
[6] Exit
**********************************************
Please select an option: 4
**********************************************

**********************************************
[1] Pounds
[2] Kilograms
**********************************************
Please enter the value of the number you are converting: 225
Please enter the unit you wish to convert from: 3
Please enter the unit you wish to convert to: 1
**********************************************

**********************************************
Please choose one of the listed choices.
**********************************************

**********************************************
Please enter the value of the number you are converting:
```

*Figure 11: Wrong-mass-conversion*

**Printing The Conversion History**

This code snippet prints out the history of all the previous conversions done with the program. As seen previously, when the conversions take place, they also save into a “history.txt” file. In this function, we see that it will print out the contents of the “history.txt" file. It does this by first opening the “history.txt” file and setting it to the history file pointer. Again, if there are errors, it will exit the program, and if not, the program will continue. Then we declare two variables, “hist[100]” and “isEmpty.” The first variable is declared to print line by line again, and the second variable is used to determine if the history file is empty or not. Afterwards, the Conversion History menu is printed, and we enter another while-loop, similar to the one we used to print the ASCII logo, but now we use it to print the contents of the “history.txt” file. However, one difference is that we set the variable isEmpty to 0. We do this because afterwards there is an if statement that will print out whether or not there is Conversion history. If there is history, then isEmpty = 0, and it will not print that, however, if the there is no history, then isEmpty will remain as 1, and print that there is no Conversion history. Lastly, it will close the history file pointer and return to the main menu.

```
void printHistory(){
    history = fopen("history.txt", "a+");
    if(history == NULL){
        printf("Error opening history file. \n");
        exit(1);
    }
    char hist[100];
    int isEmpty = 1;
    printf("\n*********************************************");
    printf("\n\t       Conversion History \n");
    printf("*********************************************\n");
    while(fgets(hist, sizeof(hist), history) != NULL){
        printf("%s", hist);
        isEmpty = 0;
    }
    if(isEmpty){
        printf("There is no Conversion History. \n");
    }
    printf("*********************************************\n");
    fclose(history);
}
```

*Figure 12: Code-to-print-history*

```
**********************************************
          The Universal Unit Converter
**********************************************
[1] Time
[2] Length
[3] Temperature
[4] Mass
[5] Print Conversion History
[6] Exit
**********************************************
Please select an option: 5
**********************************************


**********************************************
               Conversion History
**********************************************
225.00 Pound(s) is 102.04 Kilogram(s)
**********************************************

**********************************************
          The Universal Unit Converter
**********************************************
[1] Time
[2] Length
[3] Temperature
[4] Mass
[5] Print Conversion History
[6] Exit
**********************************************
Please select an option: 6
**********************************************

**********************************************
You have exited the application. All conversions have been saved in the history.txt file.
**********************************************
```

*Figure 13: Successful-conversion-history*

```
**********************************************
          The Universal Unit Converter
**********************************************
[1] Time
[2] Length
[3] Temperature
[4] Mass
[5] Print Conversion History
[6] Exit
**********************************************
Please select an option: 5
**********************************************


**********************************************
               Conversion History
**********************************************
There is no Conversion History.
**********************************************

**********************************************
          The Universal Unit Converter
**********************************************
[1] Time
[2] Length
[3] Temperature
[4] Mass
[5] Print Conversion History
[6] Exit
**********************************************
Please select an option: 6
**********************************************

**********************************************
You have exited the application. All conversions have been saved in the history.txt file.
**********************************************
```

*Figure 14: No-conversion-history*

CSE 1320

# CHAPTER THREE: CONCLUSION

## Project Weaknesses

A weakness of our unit converter is the lack of unit coverage. Because there are so many units currently in our world, our converter does not include everything. For example, a singular unit of meters can be converted at least 6 times because of the prefix system that has kilo, mega, nano, milli, etc... If we were to continue with this, each unit in both the metric and imperial measurement systems would have over one-hundred conversions, meaning that the code would have over eight-hundred lines of just conditional statements.

## Project Strengths

Some strengths of our unit converter are real-time conversions, a user-friendly interface, and that it includes conversions from both the imperial and metric systems. With our unit converter, there is no need to look up calculations to convert one's units or do any calculations, all you must do is input three things, making it very user friendly. The inputs just ask for the value of the number you have, the unit of the original value, and then the unit you want to convert to, which should instantly convert to the desired output. Also, the unit converter can go from the metric system to the imperial system or back. This is useful since there can be tricky calculations between the two systems.

## Project Enhancements

An enhancement we could make to the unit converter would be to include more units. The purpose of a unit converter is to do unit conversions, so if we are lacking a conversion that a user needs then the code loses its purpose. So, to solve this weakness, we just have to write more lines of code calculations for more unit conversions. Another enhancement would be to include some code that would automatically clear the history instead of making the user manually clear it.

## References


- Website used to create ASCII art: https://www.asciiart.eu/image-to-ascii


## Appendix

```
#include <stdio.h>
#include <stdlib.h>

void Time();
void Length();
void Temperature();
void Mass();
void printHistory();
FILE *history;

int main(){
   int selection;
   FILE *logo;
```

```
  CSE 1320
char logoString[100];
logo = fopen("logoArt.txt", "r+");
if(logo == NULL){
   printf("Error opening logo file. \n");
   exit(1);
}
while(fgets(logoString, sizeof(logoString), logo) != NULL){
   printf("%s", logoString);
}
printf("\n");
do{
   printf("\n**************************************************");
   printf("\n\t  The Universal Unit Converter \n");
   printf("**************************************************\n");
   printf("[1] Time \n");
   printf("[2] Length \n");
   printf("[3] Temperature \n");
   printf("[4] Mass \n");
   printf("[5] Print Conversion History \n");
   printf("[6] Exit \n");
   printf("**************************************************\n");
   printf("Please select an option: ");
   scanf("%d", &selection);
   printf("**************************************************\n");
   printf("\n");
   switch(selection){
   case 1:
      Time();
      break;
   case 2:
      Length();
      break;
   case 3:
      Temperature();
      break;
   case 4:
      Mass();
      break;
   case 5:
      printHistory();
      break;
   case 6:
      printf("**************************************************\n");
      printf("You have exited the application. All conversions have been saved in the history.txt file. \n");
      printf("**************************************************\n");
      break;
   default:
      printf("**************************************************\n");
      printf("Please choose one of the listed choices. \n");
```

```
      CSE 1320
         printf("************************************************\n");
         break;
      }
   }while(selection != 6);
   fclose(logo);
   return 0;
}

void printHistory(){
   history = fopen("history.txt", "a+");
   if(history == NULL){
      printf("Error opening history file. \n");
      exit(1);
   }
   char hist[100];
   int isEmpty = 1;
   printf("\n************************************************");
   printf("\n\t      Conversion History \n");
   printf("************************************************\n");
   while(fgets(hist, sizeof(hist), history) != NULL){
      printf("%s", hist);
      isEmpty = 0;
   }
   if(isEmpty){
      printf("There is no Conversion History. \n");
   }
   printf("************************************************\n");
   fclose(history);
}

void Time(){
   history = fopen("history.txt", "a+");
   if(history == NULL){
      printf("Error opening history file. \n");
      exit(1);
   }
   float unit1, unit2, num1, num2, repeat;
   printf("************************************************\n");
   printf("[1] Seconds \n");
   printf("[2] Minutes \n");
   printf("[3] Hours \n");
   printf("[4] Days \n");
   do{
      printf("************************************************\n");
      printf("Please enter the value of the number you are converting: ");
      scanf("%f", &num1);
      printf("Please enter the unit you wish to convert from: ");
      scanf("%f", &unit1);
      printf("Please enter the unit you wish to convert to: ");
```

```
  CSE 1320
   scanf("%f", &unit2);
   printf("*************************************************\n");
   if(unit1 > 4 || unit1 < 1 || unit2 > 4 || unit2 < 1 || unit1 == unit2){
      repeat = 1;
      printf("\n*************************************************\n");
      printf("Please choose one of the listed choices. \n");
      printf("*************************************************\n\n");
   } else{
      repeat = 0;
   }
}while(repeat == 1);
printf("\n*************************************************\n");
if(unit1 == 1 && unit2 == 2){
   num2 = num1/60;
   printf("%.2f Second(s) is %.2f Minute(s) \n", num1, num2);
   fprintf(history, "%.2f Second(s) is %.2f Minute(s) \n", num1, num2);
}else if(unit1 == 1 && unit2 == 3){
   num2 = num1/3600;
   printf("%.2f Second(s) is %.2f Hour(s) \n", num1, num2);
   fprintf(history, "%.2f Second(s) is %.2f Hour(s) \n", num1, num2);
}else if(unit1 == 1 && unit2 == 4){
   num2 = num1/86400;
   printf("%.2f Second(s) is %.2f Day(s) \n", num1, num2);
   fprintf(history, "%.2f Second(s) is %.2f Day(s) \n", num1, num2);
}else if(unit1 == 2 && unit2 == 1){
   num2 = num1*60;
   printf("%.2f Minute(s) is %.2f Second(s) \n", num1, num2);
   fprintf(history, "%.2f Minute(s) is %.2f Second(s) \n", num1, num2);
}else if(unit1 == 2 && unit2 == 3){
   num2 = num1/60;
   printf("%.2f Minute(s) is %.2f Hour(s) \n", num1, num2);
   fprintf(history, "%.2f Minute(s) is %.2f Hour(s) \n", num1, num2);
}else if(unit1 == 2 && unit2 == 4){
   num2 = num1/1440;
   printf("%.2f Minute(s) is %.2f Day(s) \n", num1, num2);
   fprintf(history, "%.2f Minute(s) is %.2f Day(s) \n", num1, num2);
}else if(unit1 == 3 && unit2 == 1){
   num2 = num1*3600;
   printf("%.2f Hour(s) is %.2f Second(s) \n", num1, num2);
   fprintf(history, "%.2f Hour(s) is %.2f Second(s) \n", num1, num2);
}else if(unit1 == 3 && unit2 == 2){
   num2 = num1*60;
   printf("%.2f Hour(s) is %.2f Minute(s) \n", num1, num2);
   fprintf(history, "%.2f Hour(s) is %.2f Minute(s) \n", num1, num2);
}else if(unit1 == 3 && unit2 == 4){
   num2 = num1/24;
   printf("%.2f Hour(s) is %.2f Day(s) \n", num1, num2);
   fprintf(history, "%.2f Hour(s) is %.2f Day(s) \n", num1, num2);
}else if(unit1 == 4 && unit2 == 1){
```

```
      CSE 1320
       num2 = num1*86400;
       printf("%.2f Day(s) is %.2f Second(s) \n", num1, num2);
       fprintf(history, "%.2f Day(s) is %.2f Second(s) \n", num1, num2);
    }else if(unit1 == 4 && unit2 == 2){
       num2 = num1*1440;
       printf("%.2f Day(s) is %.2f Minute(s) \n", num1, num2);
       fprintf(history, "%.2f Day(s) is %.2f Minute(s) \n", num1, num2);
    }else if(unit1 == 4 && unit2 == 3){
       num2 = num1*24;
       printf("%.2f Day(s) is %.2f Hour(s) \n", num1, num2);
       fprintf(history, "%.2f Day(s) is %.2f Hour(s) \n", num1, num2);
    }
    printf("************************************************\n");
    fclose(history);
}

void Length(){
    history = fopen("history.txt", "a+");
    if(history == NULL){
       printf("Error opening history file. \n");
       exit(1);
    }
    float unit1, unit2, num1, num2, repeat;
    printf("************************************************\n");
    printf("[1] Centimeters \n");
    printf("[2] Meters \n");
    printf("[3] Kilometers \n");
    printf("[4] Inches \n");
    printf("[5] Feet \n");
    printf("[6] Yards \n");
    printf("[7] Miles \n");
    do{
       printf("************************************************\n");
       printf("Please enter the value of the number you are converting: ");
       scanf("%f", &num1);
       printf("Please enter the unit you wish to convert from: ");
       scanf("%f", &unit1);
       printf("Please enter the unit you wish to convert to: ");
       scanf("%f", &unit2);
       printf("************************************************\n");
       if(unit1 > 7 || unit1 < 1 || unit2 > 7 || unit2 < 1 || unit1 == unit2){
          repeat = 1;
          printf("\n************************************************\n");
          printf("Please choose one of the listed choices. \n");
          printf("************************************************\n\n");
       } else{
          repeat = 0;
       }
    }while(repeat == 1);
```

```
  CSE 1320
printf("\n***********************************************\n");
if(unit1 == 1 && unit2 == 2){
   num2 = num1/100;
   printf("%.2f Centimeter(s) is %.2f Meter(s) \n", num1, num2);
   fprintf(history, "%.2f Centimeter(s) is %.2f Meter(s) \n", num1, num2);
}else if(unit1 == 1 && unit2 == 3){
   num2 = num1/100000;
   printf("%.2f Centimeter(s) is %.2f Kilometer(s) \n", num1, num2);
   fprintf(history, "%.2f Centimeter(s) is %.2f Kilometer(s) \n", num1, num2);
}else if(unit1 == 1 && unit2 == 4){
   num2 = num1/2.54;
   printf("%.2f Centimeter(s) is %.2f Inch(es) \n", num1, num2);
   fprintf(history, "%.2f Centimeter(s) is %.2f Inch(es) \n", num1, num2);
}else if(unit1 == 1 && unit2 == 5){
   num2 = num1/30.48;
   printf("%.2f Centimeter(s) is %.2f Foot/Feet \n", num1, num2);
   fprintf(history, "%.2f Centimeter(s) is %.2f Foot/Feet \n", num1, num2);
}else if(unit1 == 1 && unit2 == 6){
   num2 = num1/91.44;
   printf("%.2f Centimeter(s) is %.2f Yard(s) \n", num1, num2);
   fprintf(history, "%.2f Centimeter(s) is %.2f Yard(s) \n", num1, num2);
}else if(unit1 == 1 && unit2 == 7){
   num2 = num1/160900;
   printf("%.2f Centimeter(s) is %.2f Mile(s) \n", num1, num2);
   fprintf(history, "%.2f Centimeter(s) is %.2f Mile(s) \n", num1, num2);
}else if(unit1 == 2 && unit2 == 1){
   num2 = num1*100;
   printf("%.2f Meter(s) is %.2f Centimeter(s) \n", num1, num2);
   fprintf(history, "%.2f Meter(s) is %.2f Centimeter(s) \n", num1, num2);
}else if(unit1 == 2 && unit2 == 3){
   num2 = num1/1000;
   printf("%.2f Meter(s) is %.2f Kilometer(s) \n", num1, num2);
   fprintf(history, "%.2f Meter(s) is %.2f Kilometer(s) \n", num1, num2);
}else if(unit1 == 2 && unit2 == 4){
   num2 = num1*39.37;
   printf("%.2f Meter(s) is %.2f Inch(es) \n", num1, num2);
   fprintf(history, "%.2f Meter(s) is %.2f Inch(es) \n", num1, num2);
}else if(unit1 == 2 && unit2 == 5){
   num2 = num1*3.281;
   printf("%.2f Meter(s) is %.2f Foot/Feet \n", num1, num2);
   fprintf(history, "%.2f Meter(s) is %.2f Foot/Feet \n", num1, num2);
}else if(unit1 == 2 && unit2 == 6){
   num2 = num1*1.094;
   printf("%.2f Meter(s) is %.2f Yard(s) \n", num1, num2);
   fprintf(history, "%.2f Meter(s) is %.2f Yard(s) \n", num1, num2);
}else if(unit1 == 2 && unit2 == 7){
   num2 = num1/1609;
   printf("%.2f Meter(s) is %.2f Mile(s) \n", num1, num2);
   fprintf(history, "%.2f Meter(s) is %.2f Mile(s) \n", num1, num2);
```

```
}else if(unit1 == 3 && unit2 == 1){
   num2 = num1*100000;
   printf("%.2f Kilometer(s) is %.2f Centimeter(s) \n", num1, num2);
   fprintf(history, "%.2f Kilometer(s) is %.2f Centimeter(s) \n", num1, num2);
}else if(unit1 == 3 && unit2 == 2){
   num2 = num1*1000;
   printf("%.2f Kilometer(s) is %.2f Meter(s) \n", num1, num2);
   fprintf(history, "%.2f Kilometer(s) is %.2f Meter(s) \n", num1, num2);
}else if(unit1 == 3 && unit2 == 4){
   num2 = num1*39370;
   printf("%.2f Kilometer(s) is %.2f Inch(es) \n", num1, num2);
   fprintf(history, "%.2f Kilometer(s) is %.2f Inch(es) \n", num1, num2);
}else if(unit1 == 3 && unit2 == 5){
   num2 = num1*3281;
   printf("%.2f Kilometer(s) is %.2f Foot/Feet \n", num1, num2);
   fprintf(history, "%.2f Kilometer(s) is %.2f Foot/Feet \n", num1, num2);
}else if(unit1 == 3 && unit2 == 6){
   num2 = num1*1094;
   printf("%.2f Kilometer(s) is %.2f Yard(s) \n", num1, num2);
   fprintf(history, "%.2f Kilometer(s) is %.2f Yard(s) \n", num1, num2);
}else if(unit1 == 3 && unit2 == 7){
   num2 = num1/1.609;
   printf("%.2f Kilometer(s) is %.2f Mile(s) \n", num1, num2);
   fprintf(history, "%.2f Kilometer(s) is %.2f Mile(s) \n", num1, num2);
}else if(unit1 == 4 && unit2 == 1){
   num2 = num1*2.54;
   printf("%.2f Inch(es) is %.2f Centimeter(s) \n", num1, num2);
   fprintf(history, "%.2f Inch(es) is %.2f Centimeter(s) \n", num1, num2);
}else if(unit1 == 4 && unit2 == 2){
   num2 = num1/39.37;
   printf("%.2f Inch(es) is %.2f Meter(s) \n", num1, num2);
   fprintf(history, "%.2f Inch(es) is %.2f Meter(s) \n", num1, num2);
}else if(unit1 == 4 && unit2 == 3){
   num2 = num1/39370;
   printf("%.2f Inch(es) is %.2f Kilometer(s) \n", num1, num2);
   fprintf(history, "%.2f Inch(es) is %.2f Kilometer(s) \n", num1, num2);
}else if(unit1 == 4 && unit2 == 5){
   num2 = num1/12;
   printf("%.2f Inch(es) is %.2f Foot/Feet \n", num1, num2);
   fprintf(history, "%.2f Inch(es) is %.2f Foot/Feet \n", num1, num2);
}else if(unit1 == 4 && unit2 == 6){
   num2 = num1/36;
   printf("%.2f Inch(es) is %.2f Yard(s) \n", num1, num2);
   fprintf(history, "%.2f Inch(es) is %.2f Yard(s) \n", num1, num2);
}else if(unit1 == 4 && unit2 == 7){
   num2 = num1/63360;
   printf("%.2f Inch(es) is %.2f Mile(s) \n", num1, num2);
   fprintf(history, "%.2f Inch(es) is %.2f Mile(s) \n", num1, num2);
}else if(unit1 == 5 && unit2 == 1){
```

```
    num2 = num1*30.48;
    printf("%.2f Foot/Feet is %.2f Centimeter(s) \n", num1, num2);
    fprintf(history, "%.2f Foot/Feet is %.2f Centimeter(s) \n", num1, num2);
  }else if(unit1 == 5 && unit2 == 2){
    num2 = num1/3.281;
    printf("%.2f Foot/Feet is %.2f Meter(s) \n", num1, num2);
    fprintf(history, "%.2f Foot/Feet is %.2f Meter(s) \n", num1, num2);
  }else if(unit1 == 5 && unit2 == 3){
    num2 = num1/3281;
    printf("%.2f Foot/Feet is %.2f Kilometer(s) \n", num1, num2);
    fprintf(history, "%.2f Foot/Feet is %.2f Kilometer(s) \n", num1, num2);
  }else if(unit1 == 5 && unit2 == 4){
    num2 = num1*12;
    printf("%.2f Foot/Feet is %.2f Inch(es) \n", num1, num2);
    fprintf(history, "%.2f Foot/Feet is %.2f Inch(es) \n", num1, num2);
  }else if(unit1 == 5 && unit2 == 6){
    num2 = num1/3;
    printf("%.2f Foot/Feet is %.2f Yard(s) \n", num1, num2);
    fprintf(history, "%.2f Foot/Feet is %.2f Yard(s) \n", num1, num2);
  }else if(unit1 == 5 && unit2 == 7){
    num2 = num1/5280;
    printf("%.2f Foot/Feet is %.2f Mile(s) \n", num1, num2);
    fprintf(history, "%.2f Foot/Feet is %.2f Mile(s) \n", num1, num2);
  }else if(unit1 == 6 && unit2 == 1){
    num2 = num1*91.44;
    printf("%.2f Yard(s) is %.2f Centimeter(s) \n", num1, num2);
    fprintf(history, "%.2f Yard(s) is %.2f Centimeter(s) \n", num1, num2);
  }else if(unit1 == 6 && unit2 == 2){
    num2 = num1/1.094;
    printf("%.2f Yard(s) is %.2f Meter(s) \n", num1, num2);
    fprintf(history, "%.2f Yard(s) is %.2f Meter(s) \n", num1, num2);
  }else if(unit1 == 6 && unit2 == 3){
    num2 = num1/1094;
    printf("%.2f Yard(s) is %.2f Kilometer(s) \n", num1, num2);
    fprintf(history, "%.2f Yard(s) is %.2f Kilometer(s) \n", num1, num2);
  }else if(unit1 == 6 && unit2 == 4){
    num2 = num1*36;
    printf("%.2f Yard(s) is %.2f Inch(es) \n", num1, num2);
    fprintf(history, "%.2f Yard(s) is %.2f Inch(es) \n", num1, num2);
  }else if(unit1 == 6 && unit2 == 5){
    num2 = num1*3;
    printf("%.2f Yard(s) is %.2f Foot/Feet \n", num1, num2);
    fprintf(history, "%.2f Yard(s) is %.2f Foot/Feet \n", num1, num2);
  }else if(unit1 == 6 && unit2 == 7){
    num2 = num1/1760;
    printf("%.2f Yard(s) is %.2f Mile(s) \n", num1, num2);
    fprintf(history, "%.2f Yard(s) is %.2f Mile(s) \n", num1, num2);
  }else if(unit1 == 7 && unit2 == 1){
    num2 = num1*160900;
```

```
        printf("%.2f Mile(s) is %.2f Centimeter(s) \n", num1, num2);
        fprintf(history, "%.2f Mile(s) is %.2f Centimeter(s) \n", num1, num2);
    }else if(unit1 == 7 && unit2 == 2){
        num2 = num1*1609;
        printf("%.2f Mile(s) is %.2f Meter(s) \n", num1, num2);
        fprintf(history, "%.2f Mile(s) is %.2f Meter(s) \n", num1, num2);
    }else if(unit1 == 7 && unit2 == 3){
        num2 = num1*1.609;
        printf("%.2f Mile(s) is %.2f Kilometer(s) \n", num1, num2);
        fprintf(history, "%.2f Mile(s) is %.2f Kilometer(s) \n", num1, num2);
    }else if(unit1 == 7 && unit2 == 4){
        num2 = num1*63360;
        printf("%.2f Mile(s) is %.2f Inch(es) \n", num1, num2);
        fprintf(history, "%.2f Mile(s) is %.2f Inch(es) \n", num1, num2);
    }else if(unit1 == 7 && unit2 == 5){
        num2 = num1*5280;
        printf("%.2f Mile(s) is %.2f Foot/Feet \n", num1, num2);
        fprintf(history, "%.2f Mile(s) is %.2f Foot/Feet \n", num1, num2);
    }else if(unit1 == 7 && unit2 == 6){
        num2 = num1*1760;
        printf("%.2f Mile(s) is %.2f Yard(s) \n", num1, num2);
        fprintf(history, "%.2f Mile(s) is %.2f Yard(s) \n", num1, num2);
    }
    printf("************************************************\n");
    fclose(history);
}

void Temperature(){
    history = fopen("history.txt", "a+");
    if(history == NULL){
        printf("Error opening history file. \n");
        exit(1);
    }
    float unit1, unit2, num1, num2, repeat;
    printf("************************************************\n");
    printf("[1] Celsius \n");
    printf("[2] Fahrenheit \n");
    do{
        printf("************************************************\n");
        printf("Please enter the value of the number you are converting: ");
        scanf("%f", &num1);
        printf("Please enter the unit you wish to convert from: ");
        scanf("%f", &unit1);
        printf("Please enter the unit you wish to convert to: ");
        scanf("%f", &unit2);
        printf("************************************************\n");
        if(unit1 > 2 || unit1 < 1 || unit2 > 2 || unit2 < 1 || unit1 == unit2){
            repeat = 1;
            printf("\n************************************************\n");
```

```
      CSE 1320
         printf("Please choose one of the listed choices. \n");
         printf("************************************************\n\n");
      } else{
         repeat = 0;
      }
   }while(repeat == 1);
   printf("\n************************************************\n");
   if(unit1 == 1 && unit2 == 2){
      num2 = ((num1*9)/5) + 32;
      printf("%.2f degree(s) Celsius is %.2f degree(s) Fahrenheit \n", num1, num2);
      fprintf(history, "%.2f degree(s) Celsius is %.2f degree(s) Fahrenheit \n", num1, num2);
   }else if(unit1 == 2 && unit2 == 1){
      num2 = ((num1 - 32)*5)/9;
      printf("%.2f degree(s) Fahrenheit is %.2f degree(s) Celsius \n", num1, num2);
      fprintf(history, "%.2f degree(s) Fahrenheit is %.2f degree(s) Celsius \n", num1, num2);
   }
   printf("************************************************\n");
   fclose(history);
}

void Mass(){
   history = fopen("history.txt", "a+");
   if(history == NULL){
      printf("Error opening history file. \n");
      exit(1);
   }
   float unit1, unit2, num1, num2, repeat;
   printf("************************************************\n");
   printf("[1] Pounds \n");
   printf("[2] Kilograms \n");
   do{
      printf("************************************************\n");
      printf("Please enter the value of the number you are converting: ");
      scanf("%f", &num1);
      printf("Please enter the unit you wish to convert from: ");
      scanf("%f", &unit1);
      printf("Please enter the unit you wish to convert to: ");
      scanf("%f", &unit2);
      printf("************************************************\n");
      if(unit1 > 2 || unit1 < 1 || unit2 > 2 || unit2 < 1 || unit1 == unit2){
         repeat = 1;
         printf("\n************************************************\n");
         printf("Please choose one of the listed choices. \n");
         printf("************************************************\n\n");
      } else{
         repeat = 0;
      }
   }while(repeat == 1);
   printf("\n************************************************\n");
```

```
      CSE 1320
   if(unit1 == 1 && unit2 == 2){
      num2 = num1/2.205;
      printf("%.2f Pound(s) is %.2f Kilogram(s) \n", num1, num2);
      fprintf(history, "%.2f Pound(s) is %.2f Kilogram(s) \n", num1, num2);
   }else if(unit1 == 2 && unit2 == 1){
      num2 = num1*2.205;
      printf("%.2f Kilogram(s) is %.2f Pound(s) \n", num1, num2);
      fprintf(history, "%.2f Kilogram(s) is %.2f Pound(s) \n", num1, num2);
   }
   printf("************************************************\n");
   fclose(history);
}
```